# Augmenting the Physical Environment Through Embedded Wireless Technologies

Marco Mamei[1] and Franco Zambonelli[1]

Dipartimento di Scienze e Metodi dell'Ingegneria,
University of Modena and Reggio Emilia
Via Allegri 13, 42100 Reggio Emilia, Italy
{mamei.marco, franco.zambonelli}@unimo.it

**Abstract.** Emerging pervasive computing technologies such as sensor networks and RFID tags can be embedded in our everyday environment to digitally store and elaborate a variety of information about the surrounding. By having application agents access in a dynamic and wireless way such distributed information, it is possible to enforce a notable degree of context-awareness in applications, increase the capabilities of interacting with the physical world, and eventually give a concrete meaning to the abstract concept of agent situatedness. This paper discusses how both sensor networks and RFID tags can be used to that purpose, outlining the respective advantages and drawbacks of these technologies. Then, to ground the discussion, it presents a multiagent application for physical object tracking, facilitating the finding of "forgot-somewhere" objects in an environment.

## 1 Introduction

The never ending technological progresses in miniaturization of electronic devices and in wireless communication technologies are making possible to enrich our everyday environments (and any objects in them) with sensing, computation, and communication capabilities [1]. Overall, this may end up in an increased capability of interacting with the physical world by acquiring, in a digital form and in a wireless way, a number of information beyond the normal sensing capabilities of humans and robots, as well as in the possibility of exploiting the devices embedded in the environment as a pervasive platform for distributed computing and communication.

With reference to the multiagent systems paradigm and to agent-oriented software engineering [2], the advent of such pervasive computing technologies notably impacts on the concept of *situatedness*. Agents have always been assumed – by very definition [3] – as entities whose activities are related to some sensing and effective of the properties of some environment in which they situate for execution. However, despite this, the concept of environment has always been an overlooked topic, and a few proposals for agent languages and architectures explicitly deal with this concept in a constructive way [4]. Pervasive computing

technologies, by making available to application agents expressive digital information about the environment, can leverage the concept of situatedness from a mere conceptual definition to a practical useful feature.

Starting from the above considerations, this paper discusses which technologies can be actually used to this purpose. In particular, this paper shortly presents both sensor network technologies [1] and RFID technologies [5], and discusses how they can be exploited to augment the physical environments with the possibility of easily accessing digital information, as well as with the possibility of enforcing forms of stigmergic (i.e., environment-mediated) interactions across the physical environment. A comparative analysis of these two technologies outlines their respective advantages and limitations, and their potentials in pervasive multiagent system applications.

To ground the discussion, we presents our own experience in the implementation of a multiagent application for stigmergic physical object tracking, allowing agents (whether in the form of autonomous robots or computer-assisted humans) to find "forgot-somewhere" objects in an environment. The application relies on pheromone-based interaction, and exploit RFID tags as a physically distributed memory infrastructure in which agents can deploy pheromones and that agents can access for reading pheromone paths spread in the environment.

The rest of this paper is organized as follows. Section 2 introduces in general the concept of situatedness and the problem of interacting with physical environment. Section 3 presents and discusses sensor network technologies, while Section 4 presents and discusses RFID technologies. Section 5 presents our experience in RFID-based object tracking. Section 6 concludes and outlines open directions.

## 2 Situatedness and Physical Environments

While the concept of situatedness plays a fundamental role in the engineering of multiagent systems, the practical application of the concept cannot abstract from what actual infrastructures are available to model the environment and to interact with it. In this section, after having discussed the various facets of situatedness, we analyse how pervasive computing technologies can be used to somehow "augment" a physical environment to facilitate agents in interacting with it.

### 2.1 Computational vs. Physical Environments

Software systems are rarely developed to be deployed as stand-alone, isolated systems. Rather, in most practical cases, software systems (and multiagent systems specifically) are designed for being deployed in some sort of existing computational or physical environments, and have to necessarily interact with such environments to properly accomplish their tasks [6].

As far as computational environments are concerned, modern distributed applications are always built to interact with an existing world of data, services, and computational resources, and have to get advantage of them. For

instance, in multiagent systems for Web-based applications agents are deployed in the Web and have to mine Web data to exploit available services in order to achieve specific goals [7]. In the Grid, agents have to interact and negotiate for accessing computational and memory resources [8]. In P2P systems, networks of autonomous components (that can be assimilated to agents) have to connect and interact with each other in order to provide access to large set of shared files [9, 10].

As far as physical environments are concerned, the market is more and more demanding for a strict inter-twining of software and the physical world. Firstly, mobile computing technologies, enabling us to stay connected 24 by 7 from wherever, require context-awareness and context-dependency, to have our computer-supported activities properly adapted to the physical context and situation from which we are performing them [11]. Secondly, more and more autonomous software systems (or, which is the same, systems of autonomous robots [12]) are in need to be deployed to monitor and/or control processes occurring in the physical world, e.g., system for control of manufacturing processes [13] or of human activities [14].

### 2.2 Environment-mediated Interactions

The considerations in the above subsection justify the adoption of situatedness as a central concept in the engineering of multiagent systems. However, whether one consider computational or physical environments, the role of the environment does not simply reduce to a source of information and services, or to a set of entities that should be controlled by the multiagent system itself. Rather, the environment can also play the active central role of interaction medium, i.e., of infrastructural support for agent interactions that can occur with the active mediation of some sort of environment.

Environment-mediated interaction (aka stigmergic interaction [15]) plays an important role in nature. Indeed, the spreading and sensing of pheromones in an environment to organize the activities of ant colonies, the process of morphogenesis as enforced by diffusion of chemicals in the embryo, the movement of masses induced by gravitational fields, are all examples of stigmergic interactions [16]. In the last few years, however, stigmergic models of interactions have been recognized as very powerful to facilitate interactions in dynamic distributed systems. Indeed, stigmergic models of interactions, whether relying on synthetic pheromones, on diffusion of digital chemicals, or on spreading of virtual computational fields, are being proposed to facilitate the enforcement of adaptive interaction patterns in dynamic distributed systems and to promote self-organization and self-adaptation of activities [15, 9, 14]. Thus, the presence of some environment in which multiagent systems situates can also be exploited to support stigmergic forms of interactions.

### 2.3 Augmenting Physical Environments

In the case of agents situated in a computational environment (e.g., the Web, a P2P network, or the Grid), supporting the interaction of agents with such an environment is a rather natural process. Simply, multiagent systems are computational entities the same as the environment, and once proper data formats and interaction protocols are established, the access to the computational environment (and possibly the exploitation of such environment as an infrastructure in which to store the units of stigmergic interactions) becomes rather easy: the "sensors" and the "effectors" that the agents may use to interact reduce to a set of APIs or programming constructs.

The problem is totally different in the case of a physical environment. In this case, to access the physical environment, agents must be somehow be capable of perceiving and affecting physical properties. To this extent, an agent (whether in the form of an autonomous robot, or of an embedded controller, or of some software running on a mobile devices) must be necessarily supported by some hardware sensors and effectors to properly interact with the world.

Traditionally, most approaches for physically situated agents, assume that agents are augmented with the necessary capabilities for sensing and effecting the physical world. For instance, in the case of autonomous robots, traditional approaches assume that the robot itself is equipped with videocameras, temperature sensors, location sensors (e.g., GPS), and robotic hands. Such approach tends to notably increase the internal complexity of agents. In fact, agents not only have to perform the computational activities associated to deciding how to accomplish a goal, but have also to take care of properly internalizing and interpreting the data coming form the associated sensors, and of properly controlling their effectors to actualize their actions.

Another drawback of the above approach is that the physical environment can hardly be used to support stigmergic models of interactions, unless one adopt rather tricky solutions. If the environment is purely physical, in fact, stigmergic interactions should occur by physically affecting the properties of the environment. For example, to mimic the behavior of ants, robots would be forced to actually pollute the environment with some kind of marker, and would have to be equipped with sensor to perceive such marks [17].

The advent of pervasive computing technologies dramatically changes this scenarios. The availability of small-scale and low-cost devices that can be distributed in physical environment in a non intrusive way, that can be devoted to sense (or affect) specific properties in the environment, and that enable to interact with them in a wireless way (a capability to be easily provided to agents), enables agents to externalize all the activities devoted to interpret and control their physical activities. Simply, sensing and effecting the environment reduces in properly accessing some digital services. The result is in a notable reduction of complexity in agents, both at the hardware and at the software level.

In addition, the presence in an environment of embedded computational resources, as those that can be provided by the embedded computing devices, can be fruitfully exploited as an infrastructure to support stigmergic models

of interactions. In fact, stigmergy can take place without actually affecting the physical environment, but simply by exploiting the distributed embedded resources as stores for those data structures that are at the basis of stigmergy, e.g., pheromones, fields, etc.

Clearly, depending on the specific technologies and devices adopted, the interactions with the environment and the support of stigmergic coordination models can be more or less facilitated. In the following of this paper, we analyze in detail two different classes of devices (sensor networks and RFID tags), discuss how they can be exploited, and outline their respective advantages and drawbacks.

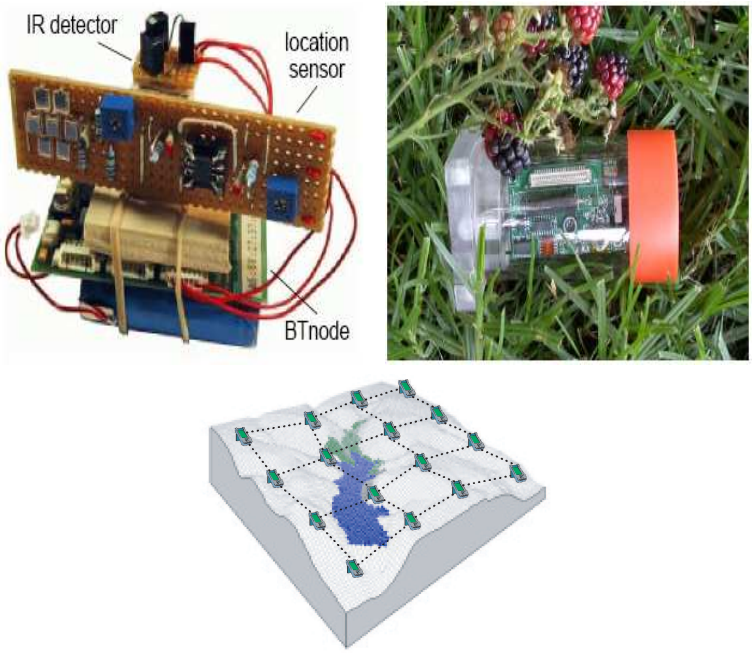


**Fig. 1.** (top) Wireless sensor devices. (bottom) sensor network in an environment

## 3 Ad-Hoc and Sensor Network

As proved in the context of the Smart Dust project at Berkeley [18, 19], it is already possible to produce fully-fledged computer-based systems of a few $cm^3$, and even much smaller ones will be produced in the next few years. Such computers, which can be enriched with communication capabilities (radio or optical), local sensing (e.g., optical, thermal, or inertial) and local effecting (e.g., opti-

cal and mechanical) capabilities, are the basic ingredients of the sensor network scenario (see Fig. 1-top).

Such a scenario implies spreading (i.e., deploying) a large number of these sensing devices across an environment, letting them create an ad-hoc wireless network by communicating with each other and perform some kind of distributed application. Traditional applications can vary from monitoring of physical parameters (e.g., monitoring weather) and distributed surveillance (e.g., tracking vehicles crossing a specific area) (see Fig. 1-bottom).

### 3.1 Deploying Digital Information

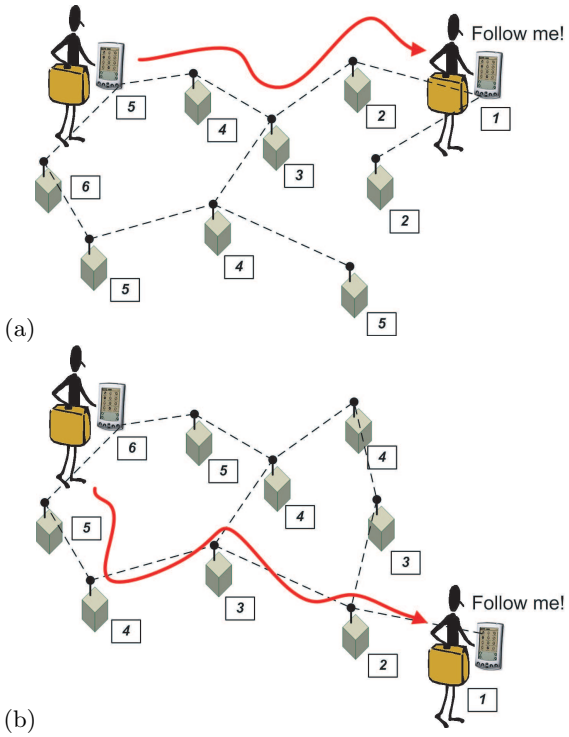


**Fig. 2.** (a) A gradient data structure enables an agent to follow another one. (b) The data structure is updated to reflect the new agent position.

In general terms, sensor networks are an ideal platform to augment the physical environment with digital information.

– Sensors can store data to represent some kind of contextual information. Moreover, they can deliver such data to agents (e.g., users with PDA) passing nearby.

– Sensors can perform computations to support and facilitate the agents' fruition to that data. For example, sensors can propagate and diffuse data across the network. They can automatically delete old and possibly corrupted information. They can combine and transform data to let it become more expressive and easy to use.

Sensors can provide agents with the data they collect to support context-awareness. For example, an agent getting an extremely-high temperature reading from a sensor nearby can infer the presence of fire and act consequently. Sensors provided with GPS devices or running a beacon-based localization algorithm [20] can provide location information. An agent getting the location of sensors nearby can infer its own actual position.

Other than providing contextual information coming from the "outside" world, sensor network can also be used to store and convey information produced by the agent themselves. Following this approach, the sensor network acts as a coordination media supporting agents' decoupled interactions and coordination [21, 22]. For example, the sensors can act as a collection of distributed tuple spaces that can be accessed by agents for the sake of enforcing coordination [23].

Moreover, relying on the sensor networking capabilities it is possible to spread distributed data structures across the environment. Data structures can be injected in the sensor network by agents and then propagate across. In addition, sensors can run maintenance algorithms to fix the data structure to account for changing conditions and dynamic networks [24].

To clarify these concepts let us focus on the problem of coordinating the movements of some autonomous agents in a distributed environment. In particular, we focus on the simple application of having two persons, provided with a PDA, moving across an environment instrumented with a sensor network infrastructure. The goal of the application is to allow one person to be guided by the PDA, to follow the other person. A simple solution based distributed data structures is the let the person to-be-followed to spread in the environment (i.e., sensor network) a data structure that increases an integer value by one at every hop as it gets farther from the source. This creates a sort of gradient that can be followed downhill by the other person to complete the application [25] (see Fig. 2(a)). If the person to-be-followed moves, it is important that the data structure adjust its shape accordingly, so that the gradient leads to that person anyway (see Fig. 2(b)). To this end, sensor nodes can run specific maintenance algorithms to keep the data structure consistent.

### 3.2 Pros and Cons

The power of this approach is that the distributed data structure provides expressive contextual information tailored for that specific task. The agent running on the PDA does not need to know any map of the environment, nor it has to execute complex algorithms to decide where to go. It just blindly follows the data structure. All the complexity of the application is moved away form from the agents and diverted into the environment-infrastructure.

Sensor networks are a powerful technology to support environment abstractions in multi-agent systems. In the long run, once current technological problems will be properly addressed, it will be the leading infrastructure of environment applications. Its main strength is that it is an *active* infrastructure: sensor nodes can run (distributed) algorithms to process data as required. For example, sensor nodes can proactively delete old information or run algorithm to aggregate data on needs. At present, however, this is also sensor network main weakness. Nodes suffer, in fact, from battery-exhaustion problems, they are costly and failure prone.

### 3.3 Related Work

A number of recent proposals address the problem of defining supporting environments for the development of adaptive, dynamic, context-aware distributed applications, suitable for pervasive computing.

The TinyLime middleware [26] proposes accessing the environmental data collected by a sensor network via an associative tuple-based mechanisms. When a user with a mobile device "walks-through" a network of distributed sensors, all the data collected by the in-range sensors automatically feeds a local tuple space of the mobile device, which thus can perceive sensorial data collected by sensors simply by reading in the local tuple space.

ObjectPlaces [27] is an interesting middleware infrastructure that offers support to exchange and share information among nodes in mobile and ad-hoc networks. Specifically, in ObjectPlaces, agents communicate indirectly through the exchange of objects that can be temporarily stored across suitable object-places (that are virtual containers stored in the ad-hoc network itself). Agents invoke operations to add and remove objects, or to observe the content of a specific object-place (via a pattern-matching process). Agents can also create object-places dynamically, and link them together to form a graph-like environment connecting related object-places.

TOTA [24] and Smart Messages [28] are two architectures for computation and communication in large networks of embedded systems. Communication is realized by sending "smart tuples" in the network, i.e., tuples which include code to be executed at each hop in the network path. These models comply with the general idea of putting intelligence in the network by letting tuples and messages execute hop-by-hop small chunk of code to determine their propagation.

Lime [29] and XMIDDLE [30] exploits transiently tuple spaces as the basis for interaction in dynamic network scenario. Each mobile device, as well as each network nodes, owns a private tuple space. Upon connection with other devices or with network nodes, the privately owned tuple spaces can merge in a federated tuple space, to be used as a common data space to exchange information.

## 4 RFID Technology

Advances in miniaturization and manufacturing have yielded postage-stamp sized radio transceivers called Radio Frequency Identification (RFID) tags that

can be attached unobtrusively to objects as small as a toothbrush. The tags are wireless and battery free. Each tag is marked with an unique identifier and provided with a tiny memory, up to some KB for advanced models, allowing to store data. Tags can be purchased off the shelf, cost roughly 0.20 Euro each and can withstand day-to-day use for years (being battery-free, they do not have power-exhaustion problems). Suitable devices, called RFID readers, can access RFID tags by radio, either for read and write operations. The tags respond or store data accordingly using power scavenged from the signal coming from the RFID reader. RFID readers divide into short- and long-range depending on the distance within which they can access RFID tags. Such distance may vary from few centimeters up to some meters. Deploying RFID technology requires that a number of places in the environment (e.g. doors, corridors, etc.) or objects (e.g. beds, washing machines, etc.) are tagged with RFID tags. Tagging a place or an object involves sticking an RFID tag on it, and making a database entry mapping the tag ID to a name. It is worth emphasizing that current trends indicate that within a few years, many household objects and furniture may be RFID-tagged before purchase, thus eliminating the overhead of tagging [31]. Moreover, some handheld devices start to be provided with RFID read and write capabilities (the Nokia 5140 phone can be already equipped with a RFID reader [32]).

### 4.1 Deploying Digital Information

The set of RFID tags deployed across the environment can be regarded as an infrastructure to store and deliver digital information.

From a general perspective, accessing the RFID tags nearby is a powerful source of context information. For example, RFID tags can reveal the location of agents in that tags can be associated to uniquely identified places. So reading the tag associated with "Prof. Smith desk" can let an agent infer its location as "Prof. Smith office". More in general, the knowledge of RFID tags (and thus objects) nearby can possibly identify a specific application context (e.g. reading a LCD-projector tag, and a microphone tag can let the agent infer of being in a meeting room).

In addition, given the fact that RFID tags can be written on-the-fly, agents can use the tags as a distributed shared memory with which to exchange information. For example, RFID tags can be accessed as if they were distributed tuple spaces [33]. A particulary significant development of this idea is related to spreading pheromone-inspired distributed data structures across the tags in the environment. The basic scenario consists of human users and robots carrying handheld computing devices, provided with a RFID reader, and running an agent-based application. The agent, unobtrusively from the user, continuously detects in range tags as the user roams across the environment. Moreover, the agent controls the RFID reader to write pheromone data structures (consisting at least in a pheromone ID) in all the tags encountered. This process creates a digital pheromone trail distributed across the tags. More formally, let us call L(t) the set of tags being sensed at time t. It is easy to see that the agent can infer that the user is moving if $L(t) \neq L(t\text{-}1)$. If instructed to spread pheromone

O, the agent will write O in all the L(t)-L(t-1) tags as it moves across the environment. For the majority of applications a pheromone trail, consisting of only an ID, is not very useful. Like in ant foraging, most applications involve agents to follow each other pheromone trails to reach the location where the agents that originally laid down the trail were directed (or, on the contrary, to reach the location where they came from). Unfortunately, an agent crossing an-only-ID-trail would not be able to choose in which direction the agent that laid down that trail was directed. From the agent point of view, this situation is like crossing a road without knowing whether to turn left or right. To overcome this problem, the agent stores in the tags also an ever increasing hop-counter associated with O - we will call this counter C(O). In particular, if an agent decides to spread pheromone O at time t, the agent reads also the counter C(O) in the L(t) set (if C(O) is not present, the agent sets C(O) to a fixed value zero). Upon a movement, the agent will store O and C(O)+1 in the tags belonging to L(t+1) that do not have O or have a lower C(O). In addition, the basic pheromone idea requires a pheromone evaporation mechanism to discard old - possibly corrupted - trails. To this end we store in the tag also a value T(O) representing the time where the pheromone O has been stored.

To read pheromones, an agent trivially accesses neighbor RFID tags reading their memories. Since RFID read operations are quite unreliable, the agent actually performs a reading cycle merging the results obtained at each iteration. Given the result, the agent will decide how to act on the basis of the perceived pheromone configuration. To realize pheromone evaporation, after reading a tag, an agent checks, for each pheromone, whether the associated timestamp is, accordingly to the agent local time, older than a certain threshold T. If it is so, the agent deletes that pheromone from the tag. This kind of pheromone evaporation leads to two key advantages:

1. Since the data space in RFID tags is severely limited, it would be most useful to store only those pheromone trails that are important for the application at a given time; old, unused pheromones can be removed.
2. If an agent does not carry its personal digital assistant or if it has been switched off, it is possible that some actions will be undertaken without leaving the corresponding pheromone trails. This cause old-pheromone trails to be possibly out-of-date, and eventually corrupted. In this context, it is of course fundamental to design a mechanism to reinforce relevant pheromones not to let them evaporate.

With this regard, an agent spreading pheromone O, will overwrite O-pheromones having an older T(O). From these considerations, it should be clear that the threshold T has to be tuned for each application, to represent the time-frame after which the pheromone is considered useless or possibly corrupted.

### 4.2 Pros and Cons

The main point in favor of this approach is its extremely low cost since it uses technologies (RFID) that are likely to be soon embedded in the scenario inde-

pendently of this application. Relying on such an implementation, a wide range of application scenarios based on pheromone interaction can be realized ranging from multi-robot coordination [34], to monitoring of human activities [35].

The main problem with this approach is related to current limitations of RFID technology. Accessing tags for reading and writing operations can fail for a number of hardly controllable issues (electromagnetic disturbances, metallic objects nearby, interferences and collisions, etc.). Moreover, in the next section, we will present and discuss some limitations in our RFID implementation of the pheromone evaporation mechanism.

### 4.3 Related Work

Several proposals, as well as ours, consider the idea of having mobile devices integrated with a RFID reader, thus having the capability of accessing RFID tags around, as sorts of digital contextual information stores. However, rather than considering the possibility of storing new information in RFID tags and enforcing coordination through them, most approaches exploit RFID tags only for reading pre-existent environmental/contextual information. For instance, the system described in [36] proposes associating location information with tags (e.g., "I am the tag of the living room") that can be read by mobile robots carrying on a RFID reader to roughly localize themselves.

The system described in [35] exploits RFID tags for inferring information about contextual activity in an environment. Users are assumed to wear an RFID reader connected with a Wi-Fi portable device so that, when the user moves and acts in the environment, the type and the sequence of tags read by the reader can suggest what the user is doing. For example, reading the tag associated to the user boss and of a video projector can let infer that the user is in a sort of important meeting with his/her boss

Pheromones spread in the environment can enable a group of users (both humans and robotics) to coordinate their respective movements. An exemplary application would be distributed environment exploration. Users could decide to explore a specific area if there are not pheromones pointing in that direction (the area is truly unexplored). In this context, it is important to remark that this approach clearly requires the presence of RFID tags before pheromones can be spread. If the environment does not contain tags at all, this approach could not be used. However, on the one hand, RFID tags are likely to be soon densely present in everywhere (embedded in tiles, bricks, furniture, etc.). On the other hand, it is possible to conceive solutions where agents physically deploy RFID tags while exploring the environment to be used for subsequent coordination. For instance, future development in plastic (and printable) RFID technology [37] let us envision the possibility of enriching an agent with a simple RFID printer to dynamically print in pavements, walls, or any type of surface, RFID tags.

## 5 Pheromone-based Object Tracking

In this section, to ground the discussion, we present a concrete application to test the introduced RFID approach. It consists in an agent-based application to easily find everyday objects (glasses, keys, etc.) forgot somewhere in our homes. The application allows everyday objects to leave virtual pheromone trails across our homes to be easily tracked afterwards.

### 5.1 Overview

Overall, the object tracking application work as follows:

- The objects to be tracked need to be tagged. For sake of clarity, we will refer to these tags as object-tags to distinguish them from the tags identifying places in the environment.
- Agents (either robotic or humans) are provided with a handheld computing device, connected to a RFID reader, and running an agent-based application.
- The agent-based application can detect, via the RFID reader, object-tags carried on by the user. Exploiting the mechanism described in the previous section, it can spread a pheromone identifying such objects into the available memory of near tags.
- This allows the object to leave a pheromone trail across the tags in the environment.
- When looking for an object, a user can instruct the agent to read in-range tags searching the object's pheromone in their memory. If such pheromone is found, the user can follow it to reach the object current location.
- Once the object has been reached, if it moves with the user (i.e. the user grabbed it), the agent automatically starts spreading again the object associated pheromone, to keep consistency with the new object location.

This application naturally suits a multi-user scenario where an user (or a robot), looking for an object moved by another user, can suddenly cross the pheromone trail the object left while moved by the other user.

### 5.2 Spreading Pheromones to Track Object

To spread pheromones, the agent needs first to understand which objects are currently being carried (i.e. moved around) by the user. To perform this task unobtrusively, it accesses the RFID reader to detect in-range RFID tags once a second. Let us call O(t) the set of object-tags being sensed at time t, L(t) the set of tags being sensed at time t. If the agent senses an object-tag O such that $O \in O(t)$, $O \notin O(t\text{-}1)$, but $L(t) \neq L(t\text{-}1)$, then the agent can infer that the user picked-up the object O and the object is moving around. In this situation, the agent has to spread O pheromone in the new location. To this end, the agent writes O in the available memory space of all the L(t) tags that do not already contain O. This operation is performed, for every object O, upon every

subsequent movement. Similarly, if the agent senses that an object-tag O $\in$ O(t-1), but O $\notin$ O(t) , then the agent infers that the user left object O. When this situation is detected the agent stops spreading O pheromone. These operations create pheromone trails of the object being moved around.

Once requested to track an object O the agent will start reading, once per second, nearby tags looking for an O-pheromone within the sensed tags L(t). If such a pheromone is found, this implies that the user crossed a suitable pheromone trail. There are two alternatives: either in L(t) there are two tags having O-pheromones with different C(O), or L(t) contains only one tag. In the former (lucky) case, the agent notifies the user about the fact it has crossed a pheromone trail and it suggest to move towards those tag having the higher C(O) . In the following, we will refer to this as grad-search, since it is like following a gradient uphill. In the latter (unlucky) case, the agent notifies the user about the fact it has crossed a pheromone trail, but nothing else. In such situation, the user has to move in the neighborhood, trying to find higher C(O) indicating the right direction to be followed (this is like dowsing – i.e. finding underground water with a forked stick – but it works!). In the following, we will refer to this as local-search. Following the agent advices, the user gets closer and closer to the object by following its pheromone trail, until reaching it.

### 5.3 Implementation and Experiments

To assess the validity of the presented approach and the effectiveness of the object tracking application, we developed a number of experiments, both adopting the real implementation and an ad-hoc simulation (to test on the large scale).

**Implementation** The real implementation consisted in tagging places and objects within our department. Overall, we tagged 100 locations within the building (doors, hallways, corridors, desks, etc.) and 50 objects (books, laptops, cd-cases, etc.). Locations have been tagged with ISO15693 RFID tags, each with a storage capacity of 512 bits (each tag contains 30 slots, 1 byte each, thus it is able to store 10 pheromones). Objects have been tagged with ISO14443B RFID tags, each with a storage capacity of 176 bits (each tag contains only the object ID) [38]. In addition, we set up three HP IPAQ 36xx running Familiar Linux 0.72 and J2ME (CVM - Personal Profile). Each IPAQ is provided with a WLAN card and a M21xH RFID reader. Each IPAQ runs the described agent-based application. Finally, a mobile robot has been realized by installing one of our wireless IPAQ (connected to a RFID reader) on a Lego Mindstorms robot (see Fig. 3).

To test on the large scale, we realized a JAVA-based simulation of the above scenario. The simulation is based on a random graph of places (each associated to a tag), and on a number of objects (each associated to an object-tag) randomly deployed in the locations-graph. Each tag has been simply realized by an array of integer values. A number of agents wanders randomly across the locations-graph collecting objects, releasing objects, and spreading pheromones accordingly. At the same time, other agents look for objects in the environment

eventually exploiting pheromone trails previously laid down by other agents. For the sake of comparison, we implemented 3 search algorithms: in blind-search, an agent explores the locations-graph disregarding pheromones. In local-search, the agent perceives the pheromones in its current node, but it cannot see the direction in which the pheromones increase. In grad-search, the agent perceives pheromones together with the directions in which they increase. The simulator, allows to perform a number of experiments changing a number of parameters such as the graph size, the number of objects, the number of agents involved, the storage capacity of the tags, etc.


**Fig. 3.** (a) Our test-bed hardware implementation. (b) Some tagged objects. (c) The Lego Mindstorms robots performing pheromone search.

**Experiments** A first group of experiments (reported in Fig. 4) aims at verifying the effectiveness of the application. Specifically, we set up two environments: one consisting of 100 tagged places with 100 objects (Fig. 4-a) and another consisting

of 2500 tagged places with 500 objects (Fig. 4-b). 10 agents populate these environments wandering around moving objects and spreading pheromones and, at the same time, looking for specific objects. In the experiments, we report the number of places visited (i.e. number of tags perceived) before finding specific objects, for different search methods, plotted over time. These results are the average of a number (over 300) of simulated experiments and verified - on a smaller scale - on the real implementation.

The more time passes the more pheromone trails get deployed. It is easy to see that blind-search does not take advantage of pheromone trails and in fact objects are found after visiting on-average half of the places. Grad-search takes a great advantage of pheromones, in fact, after several pheromone trails have been deployed, less than 10% of the places need to be visited before finding the object. Local-search is useful only in large scenarios: the time taken wandering randomly in a neighborhood, looking for the direction where a pheromone increases, hides pheromone benefits in small environments.

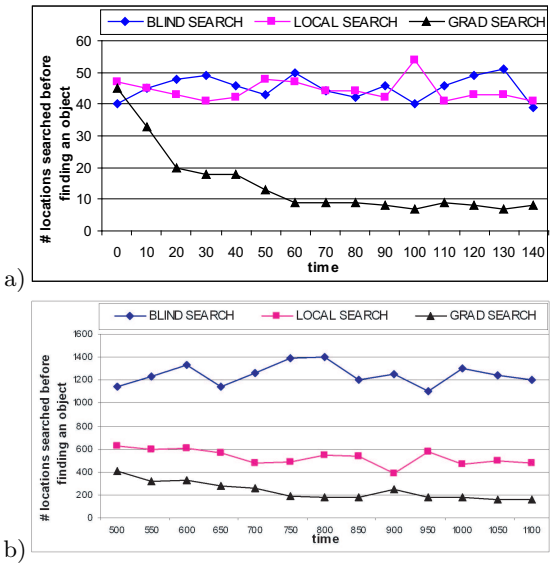


**Fig. 4.** Number of places visited before finding a specific object plotted over time. (a) 100 tagged places. (b) 2500 tagged places..

A second group of experiments aims at exploring the effects of RFID tag storage saturation upon pheromone spread. This of course represents a big problem, in fact, it can happen that pheromone trails can be interrupted, because there is not available space left on neighbor tags, while the object to be tracked moves away. This create a broken pheromone trail leading to a place that is not the actual location of the object.

In Fig. 5, we report an experiment conducted in the 100-tagged-places-environment described before. This time the tag capacity has been fixed to 50 pheromones (150 bytes), and we plot the number of places visited before finding specific objects, for different search methods, over time. Let us focus again on the grad-search behavior. It is easy to see that, when time is close to zero, grad-search works equal to blind-search, since no pheromone trails have been already laid down. After some time, grad-search works considerably better than blind-search, since pheromone trails drive agents. However, as time passes, tags capacity tend to saturate, the objects are moved, but no pheromone trails can be deployed. This situation rapidly trashes performance leading back to blind-search performance.

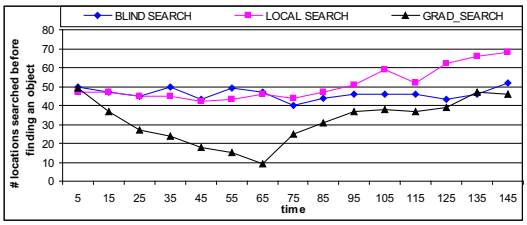


**Fig. 5.** Number of visited places before finding a specific object plotted over time, when tags tend to saturate

We get two main lessons from the above experiments.

First Lesson: in small environments grad-searches work considerably better than local-searches. However, this is not longer true in large environments, where the two methods have almost the same performance. This is clearly because the cost of "orienting" in a local neighborhood becomes negligible when the environment is large. Moreover, the drawback of grad-searches is the need for longer-range (more costly) RFID reader: the reader, in fact, must be capable of reading tags in a "one-hop" neighborhood. On the contrary local-searches can work with shorter-range (cheaper) RFID reader as well. Overall, the experiments conducted show that in near-future environments (with thousands of objects and places being tagged) local-search is a promising approach.

Second Lesson: the limited storage capacity of the RFID tags is a big problem. Basically, if the number of objects to be tracked is greater than the available slots on the RFID tag, in the long run the problem is unavoidable. Sooner or later, a new object will cross to an already full tag, breaking the pheromone trail. We still do not have a solution for this problem. Our research with regard to this topic is leading in two main directions: (i) we are currently researching more advanced pheromone evaporation mechanisms. (ii) We are considering the idea of spreading pheromone trails not only in tags but also on object-tags. The advantage would be that the more objects are in the system, the more storage

space is available for pheromones, letting the system to scale naturally. The problem is how to manage the fact that object-tags containing pheromones can be moved around, breaking the pheromone trail structure. As a partial relief from this problem, it is worth reporting that, recent RFID tags have a storage capacity in the order of the KB, making possible to track thousands of objects without changing our application.

## 6 Conclusion and Future Work

This paper presented the role of sensor network and RFID-based infrastructures to support environment abstraction in pervasive computing scenarios. These infrastructures not only allow agents to acquire context information, but also can serve as suitable media to support their coordination activities.

Our future work in this direction is twofold. On the one hand, we will try to solve technological problems related to current hardware limitations (e.g. the RFID saturation problem). On the other hand, we will try to apply environment abstractions and situatedness to several pervasive computing scenarios.

## References


1. Estrin, D., Culler, D., Pister, K., Sukjatme, G.: Connecting the physical world with pervasive networks. IEEE Pervasive Computing **1** (2002) 59 – 69
2. Zambonelli, F., Jennings, N.R., Wooldridge, M.: Developing multiagent systems: the gaia methodology. ACM Transactions on Software Engineering and Methodology **12** (2003) 417 – 470
3. Wooldridge, M., Jennings, N.R.: Intelligent agents: Theory and practice. The Knowledge Engineering Review **10** (1995) 115–152
4. Weyns, D., Parunak, V., Michel, F., Holvoet, T., Ferber, J.: Environments for Multiagent Systems, State-of-the-art and Research Challenges. Springer Verlag - LNAI 3374 (2005)
5. Want, R.: Enabling ubiquitous sensing with rfid. IEEE Computer **37** (2004) 84 – 84
6. Zambonelli, F., Parunak, H.V.D.: Signs of a revolution in computer science and software engineering. In: 3rd International Workshop on Engineering Societies in the Agents World. Volume 2577 of LNCS. Springer Verlag (2003) 13–28
7. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American (2001)
8. Foster, I., Kesselman), C.: The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco (CA) (1999)
9. Babaoglu, O., Meling, H., Montresor, A.: A framework for the development of agent-based peer-to-peer systems. In: 22nd International Conference on Distributed Computing Systems. IEEE CS Press, Vienna, Austria (2002) 15 – 22
10. Ripeani, M., Iamnitchi, A., Foster, I.: Mapping the gnutella network. IEEE Internet Computing **6** (2002) 50–57
11. Davies, N., Cheverst, K., Mitchell, K., Efrat, A.: Using and determining location in a context-sensitive tour guide. IEEE Computer **34** (2001) 35 – 41

12. Nourbakhsh, I., Sycara, K., Koes, M., Yong, M., Lewis, M., Burion, S.: Human-robot teaming for search and rescue. IEEE Pervasive Computing **4** (2005) 72 – 78
13. Bussmann, S.: Agent-oriented programming of manifacturing control tasks. In: Proceedings of the 3rd International Conference on Multi-Agent Systems. IEEE CS Press, Paris (F) (1998) 57–63
14. Mamei, M., Zambonelli, F.: Physical deployment of digital pheromones through rfid technology. In: IEEE Swarm Symposium. IEEE CS Press, Pasadena (CA), USA (2005)
15. Parunak, V.: Go to the ant: Engineering principles from natural agent systems. Annals of Operations Research **75** (1997) 69–101
16. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence. From Natural to Artificial Systems. Oxford University Press, Oxford (UK) (1999)
17. Svennebring, J., Koenig, S.: Building terrain covering ant robots: a feasibility study. Autonomous Robots **16** (2004) 313 – 332
18. Berlin, A., Gabriel, K.: Distributed mems: New challenges for computation. Computing in Science and Engineering **4** (1997) 12 – 16
19. Pister, K.: On the limits and applicability of mems technology (2000) Defense Science Study Group Report.
20. Nagpal, R., Shrobe, H., Bachrach, J.: Organizing a global coordinate system from local information on an ad hoc sensor network. In: Proceedings of the International Workshop on Information Processing in Sensor Networks. Number 2634 in LNCS. Springer-Verlag, Palo Alto, California, USA (2003)
21. Gelernter, D., N.Carriero: Coordination languages and their significance. Communication of the ACM **35** (1992) 96 – 107
22. Cabri, G., Leonardi, L., Zambonelli, F.: Engineering mobile agent applications via context-dependent coordination. IEEE Transaction on Software Engineering **28** (2002) 1040 – 1056
23. Cabri, G., Leonardi, L., Mamei, M., Zambonelli, F.: Location-dependent services for mobile users. IEEE Transactions on Systems, Man, and Cybernetics **33** (2003) 667 – 681
24. Mamei, M., Zambonelli, F.: Programming pervasive and mobile computing applications with the tota middleware. In: Proceedings of the International Conference On Pervasive Computing (Percom). IEEE CS Press, Orlando, Florida, USA (2004)
25. Mamei, M., Zambonelli, F., Leonardi, L.: Co-fields: A physically inspired approach to distributed motion coordination. IEEE Pervasive Computing **3** (2004) 52 – 61
26. Curino, C., Giani, M., Giorgetta, M., Giusti, A., Murphy, A., Picco, G.: Tinylime: Bridging mobile and sensor networks through middleware. IEEE CS Press (2005)
27. Weyns, D., Schelfthout, K., Holvoet, T.: Exploiting a virtual environment in a real-world application. In: Proceedings of the International Workshop on Environments for Multiagent Systems, Utrecht, NL (2005)
28. Borcea, C., Iyer, D., Kang, P., Saxena, A., Iftode, L.: Cooperative computing for distributed embedded systems. In: Proceedings of the International Conference on Distributed Computing Systems. IEEE CS Press, Wien, Austria (2002)
29. Picco, G., Murphy, A., Roman, G.: Lime: a middleware for logical and physical mobility. In: Proceedings of the International Conference on Distributed Computing Systems. IEEE CS Press, Providence, Rhode Island, USA (2001)
30. Mascolo, C., Capra, L., Zachariadis, Z., Emmerich, W.: Xmiddle: A data-sharing middleware for mobile computing. Wireless Personal Communications **21** (2002) 77 – 103

31. (Smart-Mobs) http://www.smartmobs.com.
32. (Nokia-Mobile-RFID-Kit) http://www.nokia.com/nokia/0,,55738,00.html.
33. Mamei, M., Quaglieri, R., Zambonelli, F.: Making tuple spaces physical with rfid tags. In: Proceedings of the Symposium on Applied Computing (SAC). ACM Press, Dijon, France (2006)
34. Payton, D., Daily, M., Estowski, R., Howard, M., Lee, C.: Pheromone robotics. Autonoumous Robots **11** (2001) 319 – 324
35. Philipose, M., Fishkin, K., Perkowitz, M., Patterson, D., Fox, D., Kautz, H., Hahnel, D.: (Inferring activities from interactions with objects)
36. Kulyukin, V., Gharpure, C., Nicholson, J., Pavithran, S.: Rfid in robot-assisted indoor navigation for visually impaired. In: Proceedings of the International Conference on Intelligent Robots and Systems. IEEE CS Press (2004)
37. Collins, G.: Next stretch for plastic electronics. Scientific American (2004)
38. (Autentiweb) http://www.autentiweb.com.